

**Progettazione di Applicazioni Web T**  
**Prova d'Esame – 21 gennaio 2022**

**Tempo a disposizione: 120 minuti**

---

La soluzione comprende la consegna elettronica dei seguenti file:

<b>Volo.zip</b>	file zip contenente il sorgente java/class, file XML e txt per il punto 1
<b>AeroportoBO.zip</b>	file zip contenente il sorgente java/class, file XML e txt per il punto 2
<b>WebRatio.zip</b>	file zip contenente il testo di risposta (txt/doc/pdf) al punto 3

**Ogni file .zip consegnato DEVE CONTENERE TUTTI e SOLI i file creati/modificati e/o ritenuti importanti in generale ai fini della valutazione (ad esempio, codice java e relativi .class, descrittori XML, file txt/doc/pdf, ecc.) e NON dell'intero progetto.**

**N.B. La prova si intende superata se il punteggio finale non è inferiore a 18 punti (su un totale di 33). In particolare, è necessario (ma non sufficiente) ottenere la sufficienza in tutti gli esercizi.**

---

**ESERCIZIO 1 - 14 punti (sufficienza: 7 punti)**

Si progetti una grammatica **XML Schema**, e un suo **documento XML** di esempio, per la modellazione delle informazioni relative al servizio prenotazioni voli aerei “**Voli**”, nel rispetto delle seguenti specifiche:

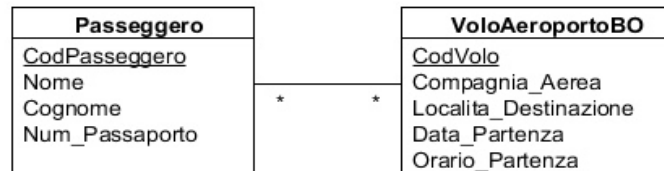
- L’archivio comprende le informazioni personali dei clienti che hanno sottoscritto il servizio.
- Le informazioni dei clienti sono costituite da nome, cognome, numero di passaporto della persona associata all’archivio e da zero o più prenotazioni.
- Ogni prenotazione è rappresentata da codice di prenotazione, nome della compagnia aerea, tipo di volo, aeroporto di partenza, data e ora di partenza, aeroporto di arrivo e data e ora di arrivo (tutti i campi devono essere modellati come obbligatori).
- Il tipo di volo può assumere uno dei seguenti valori: {“Andata-Ritorno”, “Solo-Andata”}.

Si realizzi quindi l’applicazione **Java “Voli”** che, facendo uso del **parser SAX** e del **documento XML di esempio** realizzato al punto precedente, esponga il metodo `Set<String> Voli()`, unitamente a suo un `main` di prova, in grado di restituire i codici di prenotazione di tutti i voli operati da “Alitalia” di tipo “Solo-Andata”. Stampare il risultato prodotto dal metodo `Voli()` sul file **Volo.txt**.

**Progettazione di Applicazioni Web T**  
**Prova d'Esame – 21 gennaio 2022**

**ESERCIZIO 2 - 14 punti (sufficienza: 7 punti)**

Partendo dalla realtà illustrata nel **diagramma UML** di seguito riportato, si fornisca una soluzione alla gestione della persistenza basata su tecnologia **Hibernate** in grado di “mappare” efficientemente e con uso di ID surrogati il modello di dominio rappresentato dai **JavaBean** **Passeggero** e **VoloAeroportoBO** del **diagramma UML** con le corrispondenti **tabelle relazionali** derivate dalla **progettazione logica** del **diagramma** stesso.



Nel dettaglio, dopo aver creato da applicazione Java gli **scemi delle tabelle** all'interno del proprio schema nel database **TW\_STUD** di **DB2** (esplicitando tutti i **vincoli** opportuni), **implementato i JavaBean**, definiti i **file XML di mapping** e il **file XML di properties**, si richiede **l'implementazione di un metodo in grado di restituire** “*Nome e Cognome dei passeggeri che hanno preso un volo da Bologna con destinazione Amsterdam*”.

Si crei poi un **main di prova** in grado di:

- inserire due o più tuple nelle tabelle di interesse al fine di produrre un risultato consistente rispetto a quanto richiesto dal metodo;
- fare uso corretto del metodo realizzato al punto precedente al fine di produrre la stampa del risultato, opportunamente formattata, sul file **AeroportoBO.txt**.

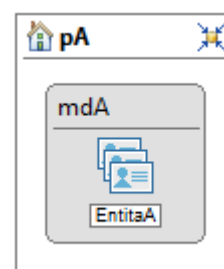
**N.B.** L'implementazione **deve limitarsi** al solo **DBMS DB2**. La soluzione **deve sfruttare il mapping N-M specificato nello schema UML**. Ogni ulteriore scelta da parte dello studente deve essere opportunamente giustificata con commenti nel codice.

**ESERCIZIO 3 - 5 punti (sufficienza: 3 punti)**

Considerando l'ipertesto progettato negli **scemi IFML di WebRatio** ai punti **a)** e **b)** di seguito riportati, il candidato descriva le tipologie di componente **Unit** (ovvero **dA** e **mdA**) racchiuse all'interno della componente pagina **pA**.



**a)**



**b)**